REMARKS

The number of inventors has been reduced to better reflect the inventorship of the present application.

The Title, Summary and Abstract have been changed.

Claims 2 – 20 have been deleted. New claims 21 – 40 have been added.

The filing fee has been calculated according to the above amendments.

Should the Examiner have any questions or comments regarding the amendments, the Examiner is invited to telephone the undersigned at the number listed below.

Respectfully submitted,

David L. McCombs
Registration No. 32,271

Date: 5-24-01
HAYNES AND BOONE, LLP
901 Main Street, Suite 3100
Dallas, Texas 75202-3789
Telephone: 214/651-5533
Facsimile: 214/651-5940
File: 26530.56
D-886948.1

EXPRESS MAIL NO.: EL828064970
DATE OF DEPOSIT: May 25, 2001

This paper and fee are being deposited with the U.S. Postal Service Express Mail Post Office to Addressee service under 37 CFR §1.10 on the date indicated above and is addressed to the Commissioner for Patents, Washington, D.C. 20231

Kathy Mettee
Name of person mailing paper and fee

Kathy Mettee
Signature of person mailing paper and fee

RE-WRITTEN SPECIFICATION MARKED UP TO SHOW ALL CHANGES PURSUANT TO 37 C.F.R. 1.121(b)

In the Title:

XML-BASED INTEGRATED SERVICES [FRAMEWORK] EVENT SYSTEM

XML-BASED INTEGRATED SERVICES EVENT SYSTEM

Inventors: Dale Lowry
969 S. 1650 E.
Springville, Utah 84663
Citizenship: USA

Samuel F. Fletcher
306 E. 2000 S.
Orem, Utah 84058
Citizenship: USA

Craig C. Johnson
3549 W. 9220 N.
Cedar Hills, Utah 84062
Citizenship: USA

Kent Sievers
432 E. 1450 N.
Orem, Utah 84097
Citizenship: USA

Assignee: Novell, Inc.
1800 South Novell Place
Provo, Utah 84606

HAYNES AND BOONE, L.L.P.
901 Main Street, Suite 3100
Dallas, Texas 75202-3789
(214) 651-5000
Haynes and Boone Docket No. 26530.56
Novell Docket No. IDR:452
D901155.1

09865257.052501

XML-BASED INTEGRATED SERVICES EVENT SYSTEM

Cross Reference

This application is a Continuation of U. S. Serial Number 09/832,970, filed April 11, 2001, which is a Continuation of U. S. Serial Number 09/741,678, filed December 19, 2000.

Background

This invention relates generally to computer software and, more specifically, to a system and method for providing distributed, directory-enabled applications using an eXtensible Markup Language ("XML") application program interface ("API") framework.

Personal computers or workstations may be linked in a computer network to facilitate the sharing of data, applications, files, and other resources. One common type of computer network is a client/server network, where some computers act as servers and others as clients. In a client/server network, the sharing of resources is accomplished through the use of one or more servers. Each server includes a processing unit that is dedicated to managing centralized resources and to sharing these resources with other servers and/or various personal computers and workstations, which are known as the "clients" of the server.

Different software applications are available through the server to the clients as network resources. The clients may also utilize "standalone" applications, which may be installed only on a client and not available through the network. The applications may perform a variety of tasks, such as word processing, email, web browsing, and many more. The applications may be written in a variety of programming languages as long as the applications are compiled to function on the underlying operating systems used by the server and the clients.

Each application is constructed using a native API that provides a set of routines, protocols, and tools. This set provides the building blocks that allow programmers to enable the applications which use the API to communicate with the operating system and other programs. Large applications such as operating systems may have hundreds of API calls to provide other applications the interfaces needed for effective communication and access to the operating system's services. Smaller applications may have a very limited set of API calls.

Because APIs are constructed for a specific application in a given programming language and targeted at a particular platform or operating system, they generally cannot be used as an interface for another application without making nontrivial modifications. In addition, such highly specific APIs make it difficult for applications to communicate if, for example, the applications were written using different programming languages or for use on different operating systems.

It is desired to provide an XML integrated services ("XIS") framework utilizing a flexible, cross-protocol, cross-language API for distributed directory-enabled applications by providing both a high level of interactivity and modular dynamic components with a common object model for both clients and servers.

Summary

In response to these and other problems, an improved system and method is provided for an event system in a distributed directory-enabled environment using an eXtensible Markup Language ("XML") application program interface. The interface includes at least one event delineated by an event parameter, where the event defines an object delineated by an object property and an object parameter.

The method defines at least one subscription filter which allows a subscriber to selectively filter the event. The event is published and a subscriber list is retrieved which includes a subscriber and the filter. The subscriber and the filter are selected from the list and the event is filtered through the filter. Subscription to the event occurs if the event passes through the filter and the subscriber may perform actions on the event after subscription. --

Brief Description of the Figures

Fig. 1 illustrates the interaction of three applications through their respective APIs.

Fig. 2 is a simple system illustrating a possible implementation of an XIS API framework.

Fig. 3 is a diagram illustrating one embodiment of an XIS API providing interaction between various applications.

Fig. 4 is an exemplary illustration of an XIS architectural framework.

Fig. 5 is a flowchart of a possible parsing sequence for one embodiment of the present invention.

Fig. 6 is a flowchart of an event sequence in an exemplary event system.

Fig. 7 is a flowchart illustrating a method by which a tag manager may resolve tag duality issues in one embodiment.

Fig. 8 is a flowchart illustrating an exemplary process for implementing thread safeness through a bridge.

Fig. 9 is a block diagram demonstrating a service performing a cross-protocol transformation.

Fig. 10 is one embodiment of a memory management scheme.

Fig. 11 is a flowchart demonstrating a basic style sheet selection sequence.

XML-BASED INTEGRATED SERVICES FRAMEWORK

Abstract

An improved system, method and software program is provided for distributed directory-enabled applications using an XML API. The improvement provides an event system, a parser, and a bridge-based object model.

The event system includes the ability to publish an event, subscribe to the event, and act on the event. The parser enables the XML API to parse XML files by accepting an XML file as an input stream, parsing the input stream, dynamically loading system services referenced in the input stream, and configuring the services. The bridge-based object model utilizes thread safeness, which enables a bridge to use semaphore access control to control thread access, smart pointers, which enable a bridge to automatically manage the memory it requires, and opaque interfaces, which allow a bridge to maintain interface compatibility when implementation changes occur in an interface.